Trabajo Práctico 8 - Herramientas de construcción de software en la nube

1- Objetivos de Aprendizaje

- Adquirir conocimientos acerca de las herramientas de integración continua en la nube.
- Configurar este tipo de herramientas.
- Implementar procesos simples de construcción automatizada.

2- Unidad temática que incluye este trabajo práctico

Este trabajo práctico corresponde a la unidad Nº: 3 (Libro Continuous Delivery: Cap 3)

3- Consignas a desarrollar en el trabajo práctico:

- Para una mejor evaluación del trabajo práctico, incluir capturas de pantalla de los pasos donde considere necesario.
- En este caso existen varias herramientas con funcionalidades similares en la nube:
  - GitHub Actions
  - CircleCI
  - Travis CI
  - Codefresh
  - Gitlab - Trial de 30 días

4- Desarrollo:

1- Pros y Contras

- Listar los pros y contras de este tipo de herramientas
- Sacar conclusiones

2- Configurando GitHub Actions

- Repetir el ejercicio 6 del trabajo práctico trabajo práctico 7 para el proyecto **spring-boot**, pero utilizando GitHub Actions.
- En GitHub, en el repositorio donde se encuentra la aplicación **spring-boot**, ir a la opción **Actions** y crear un nuevo `workflow`.
- El nombre de archivo puede ser build.xml y tendrá un contenido similar al siguiente (el path donde se encuentra el código puede ser diferente):

```yaml
# This is a basic workflow to help you get started with Actions

name: CI

# Controls when the workflow will run
on:
  # Triggers the workflow on push or pull request events but only for the
master branch
  push:
    paths:
    - 'proyectos/spring-boot/**'
    branches: [ master ]
  pull_request:
    paths:
    - 'proyectos/spring-boot/**'
    branches: [ master ]

  # Allows you to run this workflow manually from the Actions tab
  workflow_dispatch:

# A workflow run is made up of one or more jobs that can run sequentially or
in parallel
jobs:
  # This workflow contains a single job called "build"
  build:
    # The type of runner that the job will run on
    runs-on: ubuntu-latest

    # Steps represent a sequence of tasks that will be executed as part of
the job
    steps:
      # Checks-out your repository under $GITHUB_WORKSPACE, so your job can
access it
      - uses: actions/checkout@v2

      # Install Java JDK with maven
      - name: Set up JDK 8
        uses: actions/setup-java@v2
        with:
          java-version: '8'
          distribution: 'adopt'
          cache: maven

      # Compile the application
      - name: Build with Maven
        run: |
```

```
            cd proyectos/spring-boot/
            mvn -B package --file pom.xml
```

- Guardar el archivo (hacemos commit directamente en GitHub por ejemplo) y ejecutamos manualmente el pipeline.
- Explicar que realiza el pipeline anterior.

3- Utilizando nuestros proyectos con Docker

- Repetir el ejercicio 7 del trabajo práctico trabajo práctico 7, pero utilizando GitHub Actions.
- Generar `secretos` y los `pasos` necesarios para subir la imagen a Docker Hub. Referencia

4- Opcional: Configurando CircleCI

- De manera similar al ejercicio 2, configurar un build job para el mismo proyecto, pero utilizando CircleCI
- Para capturar artefactos, utilizar esta referencia: https://circleci.com/docs/2.0/artifacts/
- Como resultado de este ejercicio, subir el config.yml a la carpeta **spring-boot**

5- Opcional: Configurando TravisCI

- Configurar el mismo proyecto, pero para TravisCI. No es necesario publicar los artefactos porque TravisCI no dispone de esta funcionalidad.
- Como resultado de este ejercicio subir el archivo .travis.yml a la carpeta **spring-boot**

6- Opcional: Configurando Codefresh

- Configurar el mismo proyecto, pero para Codefresh.
- Como resultado de este ejercicio subir el archivo codefresh.yml a la carpeta **spring-boot**

7- Opcional: Configurando Gitlab

- Configurar el mismo proyecto, pero para Gitlab.
- Como resultado de este ejercicio subir el archivo .gitlab-ci.yml a la carpeta **spring-boot**